1/5/1 (Item 1 from file: 351)
DIALOG(R)File 351:Derwent WPI

012425673 **Image available**
WPI Acc No: 1999-231781/199920
XRPX Acc No: N99-171734

External data storage apparatus

Patent Assignee: SONY CORP (SONY)
Inventor: FUSE H; SASSA A

Number of Countries: 031 Number of Patents: 010

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
EP 910020	A1	19990421	EP 98118439	A	19980929	199920 B
AU 9887138	A	19990422	AU 9887138	A	19980929	199927
JP 11110141	A	19990423	JP 97267177	A	19970930	199927
CN 1220424	A	19990623	CN 98125817	A	19980930	199943
SG 71848	A1	20000418	SG 983896	A	19980925	200027
KR 99030334	A	19990426	KR 9841922	A	19980930	200028
US 6330634	B1	20011211	US 98162332	A	19980928	200204
EP 910020	B1	20021120	EP 98118439	A	19980929	200277
AU 753121	B	20021010	AU 9887138	A	19980929	200279
DE 69809527	E	20030102	DE 609527	A	19980929	200310
			EP 98118439	A	19980929	

Priority Applications (No Type Date): JP 97267177 A 19970930

Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 910020	A1	E	30	G06F-011/14	
Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI					
AU 9887138	A			G06F-009/445	
JP 11110141	A		23	G06F-003/06	
CN 1220424	A			G06F-012/00	
SG 71848	A1			G06F-003/06	
KR 99030334	A			G06F-012/00	
US 6330634	B1			G06F-012/00	
EP 910020	B1	E		G06F-011/14	
Designated States (Regional): DE FR NL					
AU 753121	B			G06F-009/445	Previous Publ. patent AU 9887138
DE 69809527	E			G06F-011/14	Based on patent EP 910020

Abstract (Basic): EP 910020 A1

NOVELTY - Boot data is stored in a number of different blocks, and an identification number indicating whether boot data in each block in which boot data has been stored, is new or old is stored in each of the blocks.

DETAILED DESCRIPTION - Data is erased in units of predetermined blocks, and one block incorporates a block in which boot data which is first read when the external storage apparatus is booted up and stored. Boot data is stored in each of the blocks in which boot data has been stored. An identification number indicating whether boot data stored in each block in which boot data has been stored is old or new, is stored in each of the blocks. An INDEPENDENT CLAIM is included for; a data processing method with which boot data is read when an external memory, arranged to erase data in units of predetermined blocks, is booted up and stored in the external memory.

USE - External storage unit with function to erase data of predetermined blocks, and processing data stored in external storage.

ADVANTAGE - Prevents occurrence of error when boot data is read.

DESCRIPTION OF DRAWING(S) - The drawing shows the structure of a memory card to which the invention is applied.

External storage apparatus (2)
Controller (11)
Flash memory unit (12)
Interface sequencer (13)
Flash memory interface sequencer (14)

Page buffer (15)
Error correction circuit (16)
Command generator (17)
pp; 30 DwgNo 2/12
Title Terms: EXTERNAL; DATA; STORAGE; APPARATUS
Derwent Class: T01; U21
International Patent Class (Main): G06F-003/06; G06F-009/445; G06F-011/14; G06F-012/00
International Patent Class (Additional): G06F-001/24; G06F-003/08; G06F-012/08; G11C-016/00; G11C-016/02
File Segment: EPI

1/5/2 (Item 1 from file: 347)
DIALOG(R)File 347:JAPIO

06168594 **Image available**
EXTERNAL STORAGE DEVICE AND DATA PROCESSING METHOD

PUB. NO.: 11-110141 A]
PUBLISHED: April 23, 1999 (19990423)
INVENTOR(s): FUSE HIROAKI
SASA SATORU
APPLICANT(s): SONY CORP
APPL. NO.: 09-267177 [JP 97267177]
FILED: September 30, 1997 (19970930)
INTL CLASS: G06F-003/06; G06F-003/08; G11C-016/02

ABSTRACT

PROBLEM TO BE SOLVED: To provide an external storage device, hardly generating an error at the time of boot data read and high in reliability, by providing plural different blocks, in which boot data are respectively stored, and providing an identification number storage part for storing an identification number showing whether the stored boot data are new or old.

SOLUTION: The plural different blocks are provided for respectively storing the boot data. Namely, the storage area of a flash memory is divided into plural blocks to be the units of data erasure. In this case, in these blocks, there are boot blocks for storing the boot data of data to be first read by a data processor 1 when a memory card 2 is started and data blocks for writing arbitrary data. Then, each block storing the boot data is equipped with the identification number storage part for storing the identification number showing whether the boot data stored in that block are new or old.

3025、23P.

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平11－110141

(43)公開日　平成11年(1999)４月23日

Corresponding to KR 1999-030334

(51)Int.Cl.[6]		識別記号	FI		
G06F	3/06	305	G06F	3/06	305A
	3/08			3/08	C
G11C	16/02		G11C	17/00	601U

審査請求　有　請求項の数６　ＯＬ　（全 23 頁）

(21)出願番号　特願平9－267177

(22)出願日　平成９年(1997)９月30日

(71)出願人　000002185
ソニー株式会社
東京都品川区北品川６丁目７番35号

(72)発明者　布施　博明
東京都品川区北品川６丁目７番35号　ソニー株式会社内

(72)発明者　佐々　哲
東京都品川区北品川６丁目７番35号　ソニー株式会社内

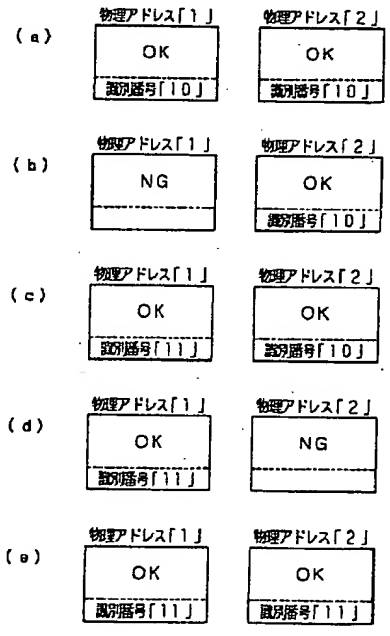
(74)代理人　弁理士　小池　晃　（外２名）

(54)【発明の名称】　外部記憶装置及びデータ処理方法

(57)【要約】

【課題】　データの消去を所定のブロック単位で行うフラッシュメモリを備えた外部記憶装置について、ブートデータ読み出し時にエラーが発生しないようにする。

【解決手段】　複数の異なるブロックにブートデータをそれぞれ格納するとともに、ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納する。そして、起動時に、上記識別番号に基づいて、複数の異なるブロックに格納された各ブートデータのうち、最新のブートデータを読み出して、当該ブートデータを用いて外部記憶装置を起動する。また、起動時に、上記識別番号に基づいて、複数の異なるブロックに格納された各ブートデータの新旧を判別し、古いブートデータがある場合には当該ブートデータを最新のブートデータに書き換える。



ブートブロックの更新時の手順

【特許請求の範囲】
【請求項１】　データの消去を所定のブロック単位で行うとともに、起動時に最初に読み出されるブートデータを格納するブロックを有する外部記憶装置であって、
複数の異なるブロックにブートデータをそれぞれ格納するとともに、
ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納することを特徴とする外部記憶装置。
【請求項２】　起動時に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータのうち、最新のブートデータが読み出され、当該ブートデータを用いて起動されることを特徴とする請求項１記載の外部記憶装置。
【請求項３】　起動時に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータの新旧が判別され、古いブートデータがある場合には当該ブートデータが最新のブートデータに書き換えられることを特徴とする請求項１記載の外部記憶装置。
【請求項４】　データの消去を所定のブロック単位で行う外部記憶装置に、当該外部記憶装置の起動時に最初に読み出されるブートデータを格納する際に、
複数の異なるブロックにブートデータをそれぞれ格納するとともに、
ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納することを特徴とするデータ処理方法。
【請求項５】　上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータのうち、最新のブートデータを読み出して、当該ブートデータを用いて外部記憶装置を起動することを特徴とする請求項４記載のデータ処理方法。
【請求項６】　上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータの新旧を判別し、古いブートデータがある場合には当該ブートデータを最新のブートデータに書き換えることを特徴とする請求項４記載のデータ処理方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、データの消去が所定のブロック単位で行われる記憶手段を備えた外部記憶装置、並びにそのような外部記憶装置に格納するデータについてのデータ処理方法に関する。
【０００２】
【従来の技術】パーソナルコンピュータやデジタルスチルカメラ等のようなデータ処理装置に用いられる外部記憶装置として、いわゆるフラッシュメモリを備えた外部記憶装置がある。
【０００３】フラッシュメモリを備えた外部記憶装置は、記憶領域を複数のブロックに分割し、データ領域の管理をブロック単位で行う。すなわち、例えば、データの消去を行う際は、ブロック単位で行う。また、例えば、記憶領域内に回復不能なエラーが生じたときには、当該エラーが生じた領域を含むブロックを使用しないようにする。なお、以下の説明では、このようなブロックのことを使用不可ブロックと称する。
【０００４】そして、このような外部記憶装置では、当該外部記憶装置にアクセスするために必要な情報等が含まれるブートデータを、予め所定のブロックに格納しておく。そして、当該外部記憶装置を起動する際は、最初にブートデータを読み出し、その後、このブートデータに基づいて当該外部記憶装置へのアクセスを行い、データの書き込みや読み出しを行う。
【０００５】このような外部記憶装置において、ブートデータには、通常、当該外部記憶装置へのアクセスに不可欠な情報が含まれており、ブートデータが読み出せなくなってしまうと、当該外部記憶装置へのアクセスが不可能となってしまう。したがって、ブートデータには、非常に高い信頼性が要求される。
【０００６】
【発明が解決しようとする課題】従来、ブートデータが格納されるブロック（以下、ブートブロックと称する。）は、予め設定された所定位置のブロックとされていた。そして、通常使用するブートブロックが使用不可ブロックとなってしまっても、ブートデータを読み出せるように、ブートデータをコピーしたものを他のブロックにも格納しておくようにしていた。そして、従来の外部記憶装置では、信頼性をより高めるために、ブートデータのコピーを多数用意し、それらをそれぞれ異なるブロックに格納しておくようにしていた。しかしながら、ブートデータのコピーを多数用意しておく手法は、記憶容量の有効利用の点では好ましくない。
【０００７】また、従来は、ブートデータが更新されたときに、ブートデータのコピー全てが正しく最新のブートデータに更新されたかを判別するようにはしていなかった。したがって、何らかの利用により、ブートデータのコピーが正しく最新のブートデータに更新されなかったりすると、最新の正しいブートデータを読み出せなくなって、外部記憶装置に対する正常なアクセスができなくなってしまうような場合があった。
【０００８】また、従来は、ブートデータのコピーを用意していたとしても、それらのコピーは、オリジナルのブートデータの読み出しに失敗したときに使用されるだけであった。したがって、従来は、オリジナルのブートデータが格納されたブートブロックの一部のビットが反転してしまったりして、オリジナルのブートデータの内容が正しくなくなっていたとしても、当該ブートデータを読み出すことさえできれば、当該ブートデータがその

まま使用されてしまうという問題があった。

【０００９】本発明は、以上のような従来の実情に鑑みて提案されたものであり、記憶領域を有効に利用しながらも、ブートデータ読み出し時にエラーが発生し難く、信頼性の高い外部記憶装置を提供することを目的としている。また、本発明は、そのような外部記憶装置を実現するデータ処理方法を提供することも目的としている。

【００１０】

【課題を解決するための手段】本発明に係る外部記憶装置は、データの消去を所定のブロック単位で行うとともに、起動時に最初に読み出されるブートデータを格納するブロックを有する外部記憶装置である。そして、複数の異なるブロックにブートデータをそれぞれ格納するとともに、ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納することを特徴としている。

【００１１】この外部記憶装置は、例えば、起動時に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータのうち、最新のブートデータが読み出され、当該ブートデータを用いて起動される。

【００１２】また、この外部記憶装置は、例えば、起動時に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータの新旧が判別され、古いブートデータがある場合には当該ブートデータが最新のブートデータに書き換えられる。

【００１３】一方、本発明に係るデータ処理方法は、データの消去を所定のブロック単位で行う外部記憶装置に、当該外部記憶装置の起動時に最初に読み出されるブートデータを格納する際に、複数の異なるブロックにブートデータをそれぞれ格納するとともに、ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納することを特徴とする。

【００１４】このデータ処理方法では、例えば、上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータのうち、最新のブートデータを読み出して、当該ブートデータを用いて外部記憶装置を起動する。

【００１５】また、このデータ処理方法では、例えば、上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータの新旧を判別し、古いブートデータがある場合には当該ブートデータを最新のブートデータに書き換える。

【００１６】

【発明の実施の形態】以下、本発明の実施の形態について、図面を参照しながら詳細に説明する。

【００１７】本発明が適用されるシステムの一例について、その全体構成を図１に示す。このシステムは、ホスト側システムとなるデータ処理装置１と、シリアルインターフェースを介してデータ処理装置１に接続される外部記憶装置であるメモリカード２とから構成される。

【００１８】なお、ここでは、データ処理装置１とメモリカード２との間でのデータのやり取りをシリアルインターフェースによって行うシステムを例に挙げるが、本発明は、データのやり取りをパラレルインターフェースによって行うシステムに対しても適用可能である。

【００１９】データ処理装置１は、演算処理装置（ＣＰＵ）３と、内部メモリ４と、補助記憶装置５と、シリアルインターフェース回路６とを備え、これらがバス７によって相互に接続されてなる。このデータ処理装置１は、例えば、補助記憶装置５に格納されているプログラムを読み出して、当該プログラムを、内部メモリ４をワークエリアとして使用して、ＣＰＵ３により実行する。このとき、必要に応じて、シリアルインターフェース回路６を介してメモリカード２との間でデータのやり取りを行う。

【００２０】なお、本発明が適用されるシステムに使用されるデータ処理装置は、外部記憶装置との間でデータのやり取りが可能なものであるならば特に限定されるものではなく、本発明は、パーソナルコンピュータ、デジタルスチルカメラ、デジタルビデオカメラ等、種々のデータ処理装置に適用可能である。

【００２１】データ処理装置１とメモリカード２とは、シリアルインターフェースによって接続されており、具体的には、少なくとも３本のデータ線ＳＣＬＫ，Ｓｔａｔｅ，ＤＩＯによって接続される。すなわち、データ処理装置１とメモリカード２とは、少なくとも、データ伝送時にクロック信号を伝送する第１のデータ線ＳＣＬＫと、データ伝送時に必要なステータス信号を伝送する第２のデータ線Ｓｔａｔｅと、メモリカード２に書き込むデータ又はメモリカード２から読み出すデータ等をシリアルに伝送する第３のデータ線ＤＩＯとによって接続され、これらを介して、データ処理装置１とメモリカード２との間でのデータのやり取りを行う。

【００２２】データ処理装置１とメモリカード２との間でのデータのやり取りは、通常、ヘッダーと実データとから構成されるファイル単位で行われる。なお、ファイルのヘッダーには、例えば、ファイルにアクセスするための情報や、データ処理装置１で実行されるプログラムで必要とされる情報等が格納される。

【００２３】メモリカード２は、図２に示すように、いわゆるコントロールＩＣからなるコントローラ１１と、コントローラ１１によって管理されるフラッシュメモリ１２とを備えている。

【００２４】コントローラ１１は、シリアル／パラレル変換やパラレル／シリアル変換等を行うシリアル／パラレル・パラレル／シリアル・インターフェース・シーケンサ１３（以下、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３と称する。）と、フラッシュメモリ１２

へのインターフェースを司るフラッシュメモリ・インターフェース・シーケンサ１４と、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３とフラッシュメモリ・インターフェース・シーケンサ１４との間でやり取りされるデータを一時的に記憶するページバッファ１５と、エラー訂正の処理を行うエラー訂正回路１６と、フラッシュメモリ１２へのアクセスを制御する制御コマンドの生成等を行うコマンドジェネレータ１７と、このメモリカード２のバージョン情報や各種属性情報等が格納されているコンフィグレーションＲＯＭ１８と、各回路に対してそれらの動作に必要なクロック信号を供給する発振器１９とを備えている。

【００２５】Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３は、少なくとも上述した３本のデータ線ＳＣＬＫ，Ｓｔａｔｅ，ＤＩＯを介して、データ処理装置１のシリアルインターフェース回路６に接続され、これらのデータ線ＳＣＬＫ，Ｓｔａｔｅ，ＤＩＯを介して、データ処理装置１との間でデータのやり取りを行う。すなわち、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３は、ページバッファ１５から送られてきたパラレルデータをシリアルデータに変換して、データ処理装置１のシリアルインターフェース回路６へ送出する。また、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３は、データ処理装置１のシリアルインターフェース回路６から送られてきたシリアルデータをパラレルデータに変換して、ページバッファ１５へ送出する。

【００２６】このＳ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３とデータ処理装置１との間でのシリアルデータの伝送は、第１のデータ線ＳＣＬＫによってデータ処理装置１から送られてくるクロック信号によって同期を取りながら、第３のデータ線ＤＩＯによって行われる。このとき、第３のデータ線ＤＩＯによってやり取りされるシリアルデータのデータ種別は、第２のデータ線Ｓｔａｔｅによって伝送されるステータス信号によって判別される。ここで、シリアルデータの種別には、例えば、フラッシュメモリ１２に書き込むべきデータ、フラッシュメモリ１２から読み出されたデータ、又はこのメモリカード２の動作を制御するための制御データ等がある。なお、ステータス信号は、メモリカード２の状態を示すためにも使用される。ステータス信号によって示されるメモリカード２の状態には、例えば、メモリカード２が何らかの処理の最中でデータ処理装置１からのデータ入力を受け付けない状態や、メモリカード２の側での処理が終了してデータ処理装置１からのデータ入力を待っている状態等がある。

【００２７】また、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３は、データ処理装置１から送られてきたデータがメモリカード２の動作を制御するための制御データである場合には、当該制御データをコマンドジェネレータ１７に送出する。

【００２８】コマンドジェネレータ１７は、データ処理装置１からＳ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３を介して送られてきた制御データに基づいて、フラッシュメモリ１２へのアクセスを制御する制御コマンドを生成し、当該制御コマンドをフラッシュメモリ・インターフェース・シーケンサ１４へ送出する。フラッシュメモリ・インターフェース・シーケンサ１４は、後述するように、この制御コマンドに基づいて、フラッシュメモリ１２にデータを書き込んだり、フラッシュメモリ１２からデータを読み出したりする。

【００２９】なお、このコマンドジェネレータ１７には、誤消去防止スイッチ２０が接続されている。そして、この誤消去防止スイッチ２０がオンになっているときには、フラッシュメモリ１２に書かれているデータを消去するように指示する制御データがデータ処理装置１から送られてきたとしても、コマンドジェネレータ１７は、フラッシュメモリ１２に書かれているデータを消去するような制御コマンドを生成しない。すなわち、このメモリカード２は、誤消去防止スイッチ２０によって、フラッシュメモリ１２に保存されているデータの消去が行えない状態と、フラッシュメモリ１２に保存されているデータの消去が行える状態とを切り換えることが可能となっている。

【００３０】Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３とフラッシュメモリ・インターフェース・シーケンサ１４との間に配されたページバッファ１５は、いわゆるバッファメモリであり、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３とフラッシュメモリ・インターフェース・シーケンサ１４との間でやり取りされるデータを一時的に記憶する。

【００３１】すなわち、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３からフラッシュメモリ・インターフェース・シーケンサ１４へ送られるデータは、先ず、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３からページバッファ１５に送られて、このページバッファ１５によって一時的に記憶される。このとき、ページバッファ１５に記憶されたデータは、エラー訂正回路１６によってエラー訂正符号が付けられる。そして、エラー訂正符号が付けられたデータは、ページバッファ１５から所定のページ単位毎（例えば１ページ＝５１２バイトとされる。）に、フラッシュメモリ・インターフェース・シーケンサ１４へと送られる。

【００３２】或いは、フラッシュメモリ・インターフェース・シーケンサ１４からＳ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３へ送られるデータは、先ず、フラッシュメモリ・インターフェース・シーケンサ１４からページバッファ１５に送られて、このページバッファ１５によって一時的に記憶される。このとき、ページバッファ１５に記憶されたデータは、エラー訂正回路１６によってエラー訂正処理が施される。そして、エラー訂

正処理が施されたデータは、ページバッファ１５から所定のページ単位毎に、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３へと送られる。

【００３３】フラッシュメモリ・インターフェース・シーケンサ１４は、コマンドジェネレータ１７からの制御コマンドに基づいて、フラッシュメモリ１２へのデータの書き込みや、フラッシュメモリ１２からのデータの読み出し等を行う。すなわち、フラッシュメモリ・インターフェース・シーケンサ１４は、コマンドジェネレータ１７からの制御コマンドに基づいて、フラッシュメモリ１２からデータを読み出して、当該データを上述のようにページバッファ１５を介して、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３へと送出する。或いは、フラッシュメモリ・インターフェース・シーケンサ１４は、コマンドジェネレータ１７からの制御コマンドに基づいて、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３からのデータを、上述のようにページバッファ１５を介して受け取り、当該データをフラッシュメモリ１２に書き込む。

【００３４】コンフィグレーションＲＯＭ１８には、このメモリカード２のバージョン情報や各種属性情報等が格納されている。コンフィグレーションＲＯＭ１８に格納された情報は、必要に応じて、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３を介してコマンドジェネレータ１７によって読み出されて使用される。すなわち、コマンドジェネレータ１７は、必要に応じて、コンフィグレーションＲＯＭ１８に格納されている情報を読み出し、この情報に基づいてメモリカード２に関する各種設定を行う。

【００３５】以上のようなメモリカード２に対して、フラッシュメモリ１２に書き込まれるデータが、上述した３本のデータ線ＳＣＬＫ，Ｓｔａｔｅ，ＤＩＯを介して、データ処理装置１からシリアルデータとして送られてくると、先ず、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３は、当該シリアルデータをパラレルデータに変換し、当該パラレルデータをページバッファ１５へ送出する。ページバッファ１５は、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３から送られてきたデータを一時的に記憶する。このとき、ページバッファ１５に記憶されたデータには、エラー訂正回路１６によってエラー訂正符号が付けられる。そして、エラー訂正符号が付けられたデータは、所定のページ単位毎にフラッシュメモリ・インターフェース・シーケンサ１４に送出される。そして、フラッシュメモリ・インターフェース・シーケンサ１４は、ページバッファ１５から送られてきたデータを、コマンドジェネレータ１７からの制御コマンドに基づいて、フラッシュメモリ１２に書き込む。以上の処理により、データ処理装置１から送られてきたデータが、フラッシュメモリ１２に書き込まれる。

【００３６】また、以上のようなメモリカード２からデータを読み出す際は、先ず、コマンドジェネレータ１７からの制御コマンドに基づいて、フラッシュメモリ・インターフェース・シーケンサ１４によって、フラッシュメモリ１２からデータが読み出される。そして、フラッシュメモリ・インターフェース・シーケンサ１４は、フラッシュメモリ１２から読み出したデータをページバッファ１５に送出する。ページバッファ１５は、フラッシュメモリ・インターフェース・シーケンサ１４から送られてきたデータを一時的に記憶する。このとき、ページバッファ１５に記憶されたデータには、エラー訂正回路１６によってエラー訂正処理が施される。そして、エラー訂正処理が施されたデータは、所定のページ単位毎にＳ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３に送出される。そして、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３は、ページバッファ１５から送られてきたデータを、シリアルデータに変換した上で、上述した３本のデータ線ＳＣＬＫ，Ｓｔａｔｅ，ＤＩＯを介して、データ処理装置１へと送出する。以上の処理により、フラッシュメモリ１２から読み出されたデータが、データ処理装置１へと送出される。

【００３７】なお、データの書き込みや読み出しを行う際は、フラッシュメモリ１２に書き込まれるデータやフラッシュメモリ１２から読み出されたデータのやり取りが行われるだけでなく、そのやり取りを制御するための制御データも、データ処理装置１からメモリカード２のＳ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３へ送られる。この制御データは、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３からコマンドジェネレータ１７に送られる。そして、コマンドジェネレータ１７は、Ｓ／Ｐ＆Ｐ／Ｓ・インターフェース・シーケンサ１３から送られてきた制御データに基づいて、フラッシュメモリ１２へのアクセスを制御する制御コマンドを生成する。そして、この制御コマンドは、フラッシュメモリ・インターフェース・シーケンサ１４に送られ、フラッシュメモリ・インターフェース・シーケンサ１４は、この制御コマンドに基づいてフラッシュメモリ１２にアクセスして、データの書き込みやデータの読み出しを行う。

【００３８】なお、メモリカード２は、上述した３本のデータ線ＳＣＬＫ，Ｓｔａｔｅ，ＤＩＯを備えるだけでなく、その他に、電圧供給用の配線や、通常は使用しないリザーブの配線等を備えていてもよい。例えば、図２並びに後掲する図３では、上述した３本のデータ線ＳＣＬＫ，Ｓｔａｔｅ，ＤＩＯの他に、４本の電源用の配線ＶＳＳ１，ＶＳＳ２，ＶＣＣ，ＩＮＴと、３本のリザーブの配線ＲＳＶ１，ＲＳＶ２，ＲＳＶ３とをメモリカード２に設けた例を挙げている。

【００３９】つぎに、以上のようなメモリカード２の具体的な外形について、図３を参照して説明する。

【００４０】メモリカード２は、合成樹脂等からなり平

面形状が長方形とされる薄肉のカード状のケース２１に、上述したコントローラ１１やフラッシュメモリ１２等が内蔵されてなる。そして、このメモリカード２は、当該メモリカード２を装着する装着機構を備えたデータ処理装置１に装着されて使用される。

【００４１】このメモリカード２のケース２１の前端部には、斜めに切り欠かれた切り欠き部２２が形成されており、更に当該切り欠き部２２が形成された部分に、１０個の凹状部２３が形成されている。そして、これらの凹状部２３の内部には、メモリカード２がデータ処理装置１の装着装置に装着されたときに、データ処理装置１の接続端子に接続される外部接続用端子が、それぞれ配されている。すなわち、このメモリカード２は、外部接続用端子として１０本の端子２４ａ，２４ｂ，２４ｃ，２４ｄ，２４ｅ，２４ｆ，２４ｇ，２４ｈ，２４ｉ，２４ｊを備えている。これらの外部接続用端子の内訳は、３本のデータ線用の端子２４ｂ，２４ｄ，２４ｈ、４本の電源用端子２４ａ，２４ｆ，２４ｉ，２４ｊ、及び３本のリザーブ端子２４ｃ，２４ｅ，２４ｇである。

【００４２】また、このメモリカード２のケース２１の上面には、誤消去防止部材２５が取り付けられている。誤消去防止部材２５は、ケース２１の内部に収納された上記誤消去防止スイッチ２０に係合されており、この誤消去防止部材２５をスライド操作することにより、誤消去防止スイッチ２０のオン／オフの切り換えを行えるようになっている。

【００４３】このメモリカード２には、データ処理装置１の装着装置に装着された際にメモリカード２がデータ処理装置１から脱落しないようにするため、ケース２０の側面の一方に円弧状の第１のロック用切欠部２６が形成され、ケース２０の側面の他方に矩形状の第２のロック用切欠部２７が形成されている。そして、このメモリカード２がデータ処理装置１の装着装置に装着されると、メモリカード２が脱落しないように、これらのロック用切欠部２６，２７が、データ処理装置１の装着装置に係合される。

【００４４】なお、図３に示したメモリカード２は、本発明が適用される外部記憶装置の一例に過ぎない。すなわち、本発明は、外部記憶装置の外形に依存することなく、どんな外形の外部記憶装置にも適用可能である。

【００４５】つぎに、以上のようなメモリカード２に搭載されるフラッシュメモリ１２の記憶領域の構造について説明する。なお、以下に説明する記憶領域の構造は、本発明が適用される外部記憶装置の記憶領域の構造の一例に過ぎない。

【００４６】すなわち、本発明は、データ消去の単位となる複数のブロックに記憶領域が分割されてなるとともに、ブートデータが格納されるブートブロックを備えた外部記憶装置に対して広く適用可能であり、その記憶領域の構造は、以下に説明するような構造でなくてもよい。

【００４７】このフラッシュメモリ１２の記憶領域は、図４（ａ）に示すように、データ消去の単位となる複数のブロックに分割されてなる。なお、これらのブロックには、このメモリカード２が起動されたときにデータ処理装置１によって最初に読み込まれるデータであるブートデータが格納されるブートブロックと、任意のデータが書き込まれるデータブロックとがある。各ブロックには、それぞれ固有の物理アドレスが付けられている。これらのブロックは、データ消去の単位であると同時に、ファイル管理上の最小単位でもある。すなわち、ファイルは１つ又は複数のブロックに格納され、１つのブロックを複数のファイルで利用することはできない。

【００４８】そして、各ブロックは、「１」又は「０」を示す２つの状態を取りうる複数のビットからなり、初期状態では、全てのビットが「１」とされており、ビット単位での変更は「１」から「０」へだけが可能となっている。すなわち、「１」及び「０」からなるデータを書き込む際、「１」については該当するビットをそのまま保持し、「０」については該当するビットを「１」から「０」に変更する。

【００４９】そして、一度書き込んだデータを消去する際は、ブロック単位で一括して初期化処理を行い、当該ブロックの全ビットを「１」とする。これにより、当該ブロックに書き込まれたデータが一括して消去され、そのブロックは再びデータの書き込みが可能な状態となる。

【００５０】なお、本発明は、上述のように各ビットが２つの状態だけを取りうるフラッシュメモリ（いわゆる２値型のフラッシュメモリ）だけでなく、各ビットが３つ以上の状態を取りうるフラッシュメモリ（いわゆる多値型のフラッシュメモリ）にも適用可能である。

【００５１】また、上記フラッシュメモリ１２の各ブロックは、図４（ｂ）に示すように、データの書き込みや読み出しの単位となる複数のページから構成される。すなわち、このフラッシュメモリ１２にデータを書き込む際は、上述したように、ページ単位にてページバッファ１５から送られてきたデータが、フラッシュメモリ・インターフェース・シーケンサ１４によってページ単位にてフラッシュメモリ１２に書き込まれる。また、このフラッシュメモリ１２からデータを読み出す際は、フラッシュメモリ・インターフェース・シーケンサ１４によってページ単位毎にデータが読み出されて、ページバッファ１５へと送られる。

【００５２】各ページは、データエリアと、冗長エリアとを有している。データエリアは、任意のデータが書き込まれる領域である。冗長エリアは、データエリアに書き込まれるデータの管理に必要な情報が格納される領域である。

【００５３】具体的には、図４（ｃ）に示すように、ブ

ロックの先頭ページの冗長エリアには、当該ブロックを管理するために必要な情報として、いわゆる分散管理情報が格納される。また、ブロックの２ページ目以降の各ページの冗長エリアにも、予備の分散管理情報として、先頭ページの冗長エリアに格納された分散管理情報と同じものが格納される。ただし、最終ページの冗長エリアには、分散管理情報ではなく、分散管理情報だけでは管理しきれない追加情報として、いわゆる追加管理情報が格納される。

【００５４】このように、このフラッシュメモリ１２では、各ブロック内の冗長エリアに分散管理情報が格納される。分散管理情報は、当該分散管理情報が格納されたブロックを管理するための情報である。この分散管理情報により、例えば、当該ブロックがファイルの先頭となるブロックであるか否かについての情報や、複数のブロックからファイルが構成される場合にはそれらのブロックの繋がりを示す情報等を得ることができる。なお、この分散管理情報については、後で詳細に説明する。

【００５５】そして、このメモリカード２では、各ブロックの分散管理情報を集めることにより、フラッシュメモリ全体を管理するための情報として、いわゆる集合管理情報を作成して、この集合管理情報をファイルとしてフラッシュメモリ１２に格納しておくようにする。

【００５６】そして、通常は、集合管理情報によって、各ブロックにアクセスするために必要な情報を得るようにする。すなわち、データ処理装置１とメモリカード２との間でデータのやり取りを行う際、データ処理装置１は、集合管理情報をメモリカード２から読み出して内部メモリ４に展開し、この集合管理情報に基づいてメモリカード２にアクセスする。これにより、データアクセスの都度、個々のブロックに格納された分散管理情報にアクセスするような必要がなくなり、より高速なデータアクセスが可能となる。

【００５７】つぎに、分散管理情報、追加管理情報、集合管理情報について、更に詳細に説明する。

【００５８】分散管理情報は、当該分散管理情報が格納されたブロックを管理するための情報であり、１６バイトの冗長エリアに書き込まれてなる。具体的には、図５に示すように、１バイトの可／不可フラグと、１バイトのブロックフラグと、４ビットの最終フラグと、４ビットの参照フラグと、１バイトの管理フラグと、２バイトの論理アドレスと、２バイトの連結アドレスと、３バイトのリザーブ領域と、２バイトの分散管理情報用エラー訂正符号と、３バイトのデータ用エラー訂正符号とからなる。

【００５９】可／不可フラグは、ブロックが使用可能状態か使用不可能状態かを示すフラグであり、具体的には、「使用可」と「使用不可」の２つの状態を示す。「使用可」は、当該ブロックが使用可能な状態を示し、「使用不可」は、当該ブロックが使用不可能な状態であることを示す。例えば、ブロック内に回復不能なエラーが生じたようなときに、この可／不可フラグが「使用不可」に設定され、当該ブロックが使用不可とされる。

【００６０】ブロックフラグは、ブロックの状態を示すフラグであり、具体的には、「未使用」「先頭使用」「使用」「未消去」の４つの状態を示す。「未使用」は、当該ブロックが未使用又は消去済みで、初期状態（全ビットが「１」の状態）とされており、直ぐにデータの書き込みが可能な状態を示す。「先頭使用」は、当該ブロックがファイルの先頭で使用されている状態を示す。なお、ブートデータが格納されたブートブロックにおいて、ブロックフラグは「先頭使用」とされる。「使用」は、当該ブロックがファイルの先頭以外で使用されている状態を示す。ブロックフラグが「使用」のとき、当該ブロックは、他のブロックから連結されていることとなる。「未消去」は、当該ブロックに書かれていたデータが不要となった状態を示す。例えば、データの消去を行うときに、取りあえずブロックフラグを「未消去」にしておき、処理時間に余裕があるときに、ブロックフラグが「未消去」になっているブロックを消去するようにする。これにより、消去処理をより効率良く行うことが可能となる。

【００６１】最終フラグは、ファイルが終わっているか否かを示すフラグであり、具体的には、「ブロック連続」「ブロック最終」の２つの状態を示す。「ブロック連続」は、次のブロックへの連結があることを示す。すなわち、「ブロック連続」は、当該ブロックに格納されたファイルにはまだ続きがあり、当該ファイルが他のブロックに続いていることを示す。「ブロック最終」は、最終ブロックであることを示す。すなわち、「ブロック最終」は、当該ブロックに格納されたファイルが、このブロックで終了していることを示す。

【００６２】参照フラグは、追加管理情報の参照を指定するためのフラグであり、具体的には、「参照情報なし」「参照情報あり」の２つの状態を示す。「参照情報なし」は、ブロックの最終ページの冗長領域に、有効な追加管理情報が存在しないことを示す。「参照情報あり」は、ブロックの最終ページの冗長領域に、有効な追加管理情報が存在していることを示す。

【００６３】管理フラグは、ブロックの属性等を示すフラグである。例えば、この管理フラグによって、当該ブロックが読み出し専用ブロックか、或いは書き込みも可能なブロックであるかが示される。また、例えば、この管理フラグによって、当該ブロックがブートブロックであるか、或いはデータブロックであるかが示される。

【００６４】論理アドレスは、文字通りそのブロックの論理アドレスを示す。この論理アドレスの値は、データの書き換えを行うときなどに必要に応じて更新される。なお、論理アドレスの値は、正常に処理が行われている限り、同じ論理アドレスの値を同時に複数のブロックが

持つことがないように設定される。

【００６５】連結アドレスは、当該ブロックに連結するブロックの論理アドレスを示す。すなわち、ブロックに格納されたファイルにはまだ続きがあり、当該ファイルが他のブロックに続いている場合、連結アドレスには、そのファイルの続きが格納された次のブロックの論理アドレスの値が設定される。

【００６６】分散管理情報用エラー訂正符号は、分散管理情報のうち、管理フラグ、論理アドレス、連結アドレス及びリザーブ領域に書き込まれデータを対象としたエラー訂正符号である。なお、可／不可フラグ、ブロックフラグ、最終フラグ及び参照フラグは、分散管理情報用エラー訂正符号によるエラー訂正の対象となっていない。したがって、可／不可フラグ、ブロックフラグ、最終フラグ及び参照フラグは、分散管理情報用エラー訂正符号を更新することなく書き換えることが可能となっている。

【００６７】データ用エラー訂正符号は、当該データ用エラー訂正符号が格納されているページのデータエリアに書き込まれたデータを対象としたエラー訂正符号である。

【００６８】なお、分散管理情報用エラー訂正符号やデータ用エラー訂正符号は、メモリカード２の内部に配されたエラー訂正回路１６によって使用される。したがって、これらのエラー訂正符号を用いてのエラー訂正は、データ処理装置１に依存することなく、メモリカード２に依存した任意の手法を使用することができる。

【００６９】追加管理情報は、ブロックの最終ページの１６バイトの冗長エリアに格納される情報であり、分散管理情報だけでは管理しきれない追加情報を含んでいる。

【００７０】具体的には、追加管理情報は、図６に示すように、１バイトの可／不可フラグと、１バイトのブロックフラグと、４ビットの最終フラグと、４ビットの参照フラグと、１バイトの識別番号と、２バイトの有効データサイズと、５バイトのリザーブ領域と、２バイトの追加管理情報用エラー訂正符号と、３バイトのデータ用エラー訂正符号とからなる。

【００７１】ここで、可／不可フラグ、ブロックフラグ、最終フラグ、参照フラグ、リザーブ領域及びデータ用エラー訂正符号については、分散管理情報の場合と同様である。また、追加管理情報用エラー訂正符号は、分散管理情報における分散管理情報用エラー訂正符号に相当するものであり、追加管理情報のうち、識別番号、有効データサイズ及びリザーブ領域に書き込まれデータを対象としたエラー訂正符号である。

【００７２】そして、識別番号及び有効データサイズとが、分散管理情報だけでは管理しきれない追加情報として、追加管理情報に含まれている。

【００７３】識別番号は、エラー処理用の情報であり、ブロックのデータを書き換える度に、この識別番号の値がインクリメントされる。この識別番号は、何らかのエラーが発生して、同じ論理アドレスを持つブロックが複数存在するようになってしまった場合に、それらのブロックに書き込まれたデータの新旧を識別するために使用される。なお、識別番号には１バイトの領域が使用され、その値の範囲は「０」から「２５５」までであり、その初期値は「０」とされる。なお、識別番号が「２５５」を越えたときには「０」に戻される。そして、同じ論理アドレスを持つデータブロックが複数存在する場合には、この識別番号の値が小さい方のデータブロックを有効とする。また、後で詳細に説明するように、同じ論理アドレスを持つブートブロックが複数存在する場合には、この識別番号の値が大きい方のブートブロックを有効とする。

【００７４】また、有効データサイズは、ブロック内の有効なデータのサイズを示す。すなわち、当該ブロックのデータエリアに空きがある場合、有効データサイズには、当該データエリアに書き込まれたデータのサイズを示す値が設定される。このとき、分散管理情報の参照フラグは「参照情報あり」に設定される。なお、ブロックのデータエリアに空きがない場合、有効データサイズには、当該データエリアに空きがないことを示す値として、「0xffff」が設定される。

【００７５】なお、以上のような分散管理情報及び追加管理情報は、ブロック内のデータが更新される毎に、常に最新情報となるように更新される。

【００７６】集合管理情報は、上述したように、各ブロックの分散管理情報を集めて作成されてなる情報であり、ファイルとしてフラッシュメモリ１２に格納される。すなわち、図７に示すように、各ブロックの分散管理情報から、全ブロックをまとめて管理するための情報である集合管理情報のファイルが作成され、この集合管理情報が所定のブロックのデータエリアに格納される。なお、集合管理情報は、１つのブロックに格納されるものであっても、複数のブロックにわたって格納されるものであってもよい。そして、データ処理装置１は、通常は、この集合管理情報によって、各ブロックにアクセスするために必要な情報を得るようにする。

【００７７】このメモリカード２の起動時には、先ず、データ処理装置１によって、ブートブロックからブートデータが読み込まれるが、このブートデータには、集合管理情報が格納されているブロックの物理アドレスが指定されている。そこで、データ処理装置１は、当該物理アドレスに対応したブロックに格納されている集合管理情報を読み出して内部メモリ４に展開し、この集合管理情報に基づいてメモリカード２にアクセスする。なお、フラッシュメモリ１２から集合管理情報を正常に読み出せなかった場合、データ処理装置１は、全てのブロックの分散管理情報を読み込み、集合管理情報を再構築して

使用する。

【００７８】そして、データ処理装置１は、データの書き換え等を行う毎に、内部メモリ４に展開した集合管理情報を、フラッシュメモリ１２の実際の状態と整合するように（すなわち、分散管理情報の内容と整合するように）、随時更新していく。一方、フラッシュメモリ１２にファイルとして格納されている集合管理情報は、データの書き換え等を行う毎に更新するのではなく、適当なタイミングにて、その変更内容が一括して更新される。なお、ここでの適当なタイミングとは、例えば、電源を落とす前や、メモリカード２へのアクセスが所定時間以上なされなかったときや、データの書き換えが所定回数以上行われたときなどである。

【００７９】一般にフラッシュメモリ１２の書き換え可能回数には上限があるが、このようにフラッシュメモリ１２にファイルとして格納されている集合管理情報の書き換えをある程度まとめて一括して行うようにすることで、集合管理情報が格納されているブロックの書き換え回数を削減することができ、メモリカード２の長寿命化を図ることができる。

【００８０】つぎに、本発明のポイントとなるブートブロックの取り扱いについて詳細に説明する。

【００８１】ブートデータは、メモリカード２を起動したときにデータ処理装置１によって最初に読み込まれるデータであり、ブートデータには、メモリカード２にアクセスするために必要不可欠な情報が含まれている。したがって、ブートデータが格納されたブートブロックにエラーが生じて、ブートデータを正常に読み出すことができなくなると、メモリカード２へのアクセスが不可能になってしまう。したがって、ブートブロックには、非常に高い信頼性が要求される。

【００８２】そこで、本発明を適用したメモリカード２では、常にブートデータをフラッシュメモリ１２の先頭の有効な２ブロックにそれぞれ格納しておくようにする。なお、「有効なブロック」とは、使用可能状態にあるブロックのことである。換言すれば、「有効なブロック」とは、可／不可フラグが「使用可」に設定されているブロックである。すなわち、ブートデータをブロックに格納する際に、有効でないブロック（すなわち、可／不可フラグが「使用不可」に設定されているブロック）は、使用することなく飛ばして、有効なブロックにブートデータを格納してブートブロックとする。

【００８３】このように、本発明を適用したメモリカード２では、固定された特定のブロックをブートブロックにするのではなく、有効でないブロックが生じたときには、ブートデータを格納するブロックを変更して、有効な２つのブロックに常にブートデータが格納されるようにする。すなわち、例えば、メモリカード２の先頭のブロックにエラーが生じたとしても、常にブートデータは有効なブロックに二重に保持される。したがって、本発明を適用したメモリカード２の信頼性は非常に高いものとなる。

【００８４】また、本発明を適用したメモリカード２では、追加管理情報に含まれている識別番号を、データブロックの場合とブートブロックの場合とで区別して使用する。

【００８５】すなわち、データブロックの場合には、同じ論理アドレスのブロックが複数あったときに、それらのブロックに格納されているデータの新旧を識別するために識別番号を使用する。ここで、同じ論理アドレスを持つデータブロックが複数存在するのは、多くの場合、データブロック更新時に何らかのエラーが発生したためであり、このとき、識別番号の値が小さい方のデータブロックに格納されているデータが更新前のデータである。そこで、同じ論理アドレスを持つデータブロックが複数存在する場合には、データを更新する前の状態に戻すために、識別番号の値が小さい方のデータブロックを選択し、当該データブロックに格納されているデータを有効なデータとして採用する。

【００８６】なお、データブロックの識別番号は、データブロックに対して論理アドレスが新規に割り当てられたときに初期化されて「０」とされる。また、この識別番号は、データブロックの更新時に１インクリメントされる。なお、上述したように、識別番号が「２５５」を越えた場合には「０」に戻される。

【００８７】一方、ブートブロックの場合には、上述したようにフラッシュメモリ１２の先頭の有効な２ブロックに格納されたブートデータについて、それらの新旧を識別するために識別番号を使用する。ブートデータは、メモリカード２へのアクセスに必要不可欠な情報であり、常に最新の情報を用いる必要がある。そして、２つのブートブロックの識別番号が異なるのは、多くの場合、ブートブロックの更新時に何らかのエラーが発生したためであり、このとき、識別番号の値が大きいブートブロックに格納されているブートデータが、より新しい情報を含むブートデータである。そこで、２つのブートブロックについて、それらの識別番号が異なる場合には、識別番号の値が大きい方のブートブロックを選択し、当該ブートブロックに格納されているデータをブートデータとして採用する。

【００８８】なお、このメモリカード２は、最初に使用するときにフォーマットされるが、このフォーマット時に、ブートブロックの識別番号は初期化され「０」とされる。そして、この識別番号は、ブートブロックに格納されているブートデータを更新した時に１インクリメントされる。なお、ブートブロックの場合も、データブロックのときと同様に、識別番号が「２５５」を越えた場合には「０」に戻される。

【００８９】つぎに、以上のようにフラッシュメモリ１２の先頭の有効な２ブロックをブートブロックにすると

ともに、それらのブートブロックの新旧を識別番号を用いて管理する際の手順について、更に詳細に説明する。

【００９０】（１）フォーマット時の手順

上述したように、このメモリカードは、最初に使用するときにフォーマットされる。そして、このフォーマット時にブートブロックが作成される。そこで、まず、フォーマット時のブートブロック作成の手順について説明する。

【００９１】フォーマット時に、データ処理装置１は、先ず、メモリカード２のコンフィグレーションＲＯＭ１８に格納されている情報を読み出し、この情報等に基づいてブートブロックに格納するデータ（すなわちブートデータ）を作成する。なお、このデータには、当該データを読み出したときに、そのデータがブートデータであることが分かるように、当該データがブートデータであることを示す情報（以下、ブート識別子と称する。）も含めておく。

【００９２】次に、作成したブートデータを、フラッシュメモリ１２の先頭の２ブロック（すなわち、物理アドレスの値が最も小さいブロックと、物理アドレスの値がその次に小さいブロック）にそれぞれ書き込み、ブートブロックと予備のブートブロックとを作成する。このとき、ブートデータが書き込まれるブロックの追加管理情報の識別番号を「０」に設定する。エラーなく終了すれば、ブートブロックの作成を終了する。

【００９３】ここでのブートデータの書き込みは、具体的には、先ず、該当するブロックに対して消去処理を施し、その後、当該ブロックに対してブートデータを書き込むことによって行う。このとき、当該ブロックに対して消去処理が正常に行えなかった場合や、消去処理が施されたブロックに対してブートデータを正常に書き込めなかった場合には、当該ブロックの可／不可フラグが「使用不可」に設定され、当該ブロックは使用不可状態とされる。このとき、可／不可フラグを「使用不可」に設定できなかった場合には、可／不可フラグを「使用不可」に設定する処理を何度かリトライする。それでも設定できない場合には、データ処理装置１はメディア異常と判断し、そのメモリカード２を受け付けないようにする。なお、消去処理が施されたブロックに対してブートデータを正常に書き込めなかった場合には、当該ブロックに途中まで書き込んだブートデータが残ってしまったりするようなことがないように、可／不可フラグを「使用不可」に設定する前に、当該ブロックに対して消去処理を施しておく。

【００９４】そして、本発明を適用したメモリカード２では、ブートブロックにしようとしたブロックにエラーが発生した場合には、次に使用可能なブロックに対して、同様にブートデータの書き込みを試みる。そして、正常で且つ内容が同一のブートブロックが２個作成されるまで、この処理を続ける。これにより、内容が同一のブートデータがフラッシュメモリ１２の先頭の有効な２ブロックにそれぞれ格納され、先頭の有効な２ブロックがブートブロック及び予備のブートブロックとなる。

【００９５】ただし、ブートデータの書き込みは、対象となるブロックの物理アドレスが所定の値Ｍになったら、内容が同一のブートブロックが２個作成されていなくても、処理を終了する。この所定の値Ｍは、フラッシュメモリ１２の特性等に応じて予め規定しておく。そして、対象となるブロックの物理アドレスが所定の値Ｍになったら、ブートブロックが１つだけしか作成されていなくても、ブートブロックの作成を終了する。また、対象となるブロックの物理アドレスが所定の値Ｍになっても、ブートブロックが１つも作成されない場合には、メディア異常と判断し、そのメモリカード２を受け付けないようにする。

【００９６】ところで、一般にフラッシュメモリ１２は、書き換え回数が多いブロックでエラーが発生する可能性が高く、一旦エラーが出始めると、その後もエラーが生じる確率が高い。そこで、以前ブートブロックとして使用されたが、エラーになって使用不可になったブロックは、たとえそのエラーが一時的なものであったとしても、そのブロックの再利用はしないようにする。このような規則を採用することにより、非常に重要な情報であるブートデータは、より信頼性の高いブロックに格納されることとなり、ブートブロックの信頼性の向上を図ることができる。なお、この規則は、先頭のブロックから２つめのブートブロックまでの間にだけ、或いは２つめのブートブロックが無い場合には物理アドレスが上記所定の値Ｍに至るまでの間にだけ適用する。すなわち、この規則は、高い信頼性が要求されるブートブロックについてだけ適用し、それ以降のブロック（すなわちデータブロック）に対しては適用しない。これにより、記憶領域のより有効な利用を図ることができる。

【００９７】なお、ブートブロックの読み込み時にエラーが発生した場合、当該ブートブロックを直ぐに使用不可にするようなことはしない。例えば、ブートブロックの書き換え中にメモリカード２がデータ処理装置１から抜かれた場合、ブートブロックの書き換えが中途半端に終了している可能性が高い。このようなとき、そのブートブロックを次に読み込んだ場合にエラーが発生するが、そのブロックは再利用が可能であるので、当該ブロックを使用不可にする必要はない。

【００９８】しかしながら、データ処理装置１は、このような読み込みエラーと、フラッシュメモリ１２で回復不能な障害が発生したために起きた読み込みエラーとを区別して検出することができない。そこで、ブートブロックの読み込み時にエラーが発生した場合は、当該ブートブロックを消去して改めてブートデータを書き込む。もし、フラッシュメモリ１２で回復不能な障害が発生している場合は、ブロック消去やデータ書き込み時にもエ

ラーが発生することとなる。そこで、このようにブロック消去やデータ書き込み時にエラーが生じたときにだけ、そのブロックを使用不可とする。

【００９９】（２）ブートブロック更新時の手順

ブートデータは、常に同じデータとは限らず、メモリカード２の使用に伴い、その内容を変更する必要が生じる。したがって、ブートブロックを更新して、ブートデータを書き換える場合がある。このブートデータの書き換えは、以下に示すルールに基づいて行う。

【０１００】・ブートデータの書き換えを行うようなときには、集合管理情報の内容を変更する必要がある。そこで、ブートデータの書き換えを行う前に、フラッシュメモリ１２にファイルとして格納されている集合管理情報を無効にする。なお、次に集合管理情報を用いるときは、改めて分散管理情報から構築し直す。

【０１０１】・ブートデータの内容を書き換える際は、現在ブートデータが格納されているブートブロックに対して消去処理を施し、そのブロックに新たなブートデータを書き込む。すなわち、ブートデータを書き換えるときは、別のブロックを使用することなく、同一ブロックの内容を更新する。ただし、ブートブロックの更新時にエラーが発生し、当該ブロックが使用不可になった場合は、この限りではない。

【０１０２】・新しいブートデータに書き換える場合は、上述したように２つ作成したブートブロックのうち、物理アドレスが小さい方から行う。このとき、内容を更新したブートブロックの識別番号を１インクリメントする。ただし、書き換え前の識別番号の値が「２５５」の場合は「０」とする。

【０１０３】・予備のブートブロック（すなわち物理アドレスの大きいほうのブートブロック）を更新して、もうひとつのブートブロックと同一の内容となるように、新しいブートデータに書き換えたときは、当該予備のブートブロックの識別番号を、もうひとつのブートブロックの識別番号と同じ値にする。

【０１０４】以上のようなルールに基づいて、ブートブロックを更新する際の手順について、図８を参照して説明する。

【０１０５】なお、図８では、物理アドレス「１」のブロックが通常使用されるブートブロック（以下、第１のブートブロックと称する。）とされ、物理アドレス「２」のブロックが予備のブートブロック（以下、第２のブートブロックと称する。）とされている例を示している。また、図８では、ブートデータが正常に書き込まれており、当該ブートデータの読み出しが可能な状態のことを「ＯＫ」として示している。また、ブートデータ書き換えの最中等により、ブートデータの読み出しが不可能になっている状態のことを「ＮＧ」として示している。

【０１０６】図８（ａ）は、ブートブロック書き換え前の正常な状態を示している。このとき、第１のブートブロックの識別番号と、第２のブートブロックの識別番号とは、同じ値である。具体的には、図８（ａ）の例では、第１のブートブロックの識別番号が「１０」であり、第２のブートブロックの識別番号も同じ「１０」とされている。

【０１０７】ブートブロックを更新する際は、先ず、図８（ｂ）に示すように、物理アドレスの小さいほうのブートブロック、すなわち第１のブートブロックのほうから、ブートデータの書き換えを行う。このとき、第１のブートブロックは、ブートデータの書き換えの最中であるので、ブートブロックとしては使用できない「ＮＧ」状態となる。

【０１０８】もしもこの段階で、メモリカード２がデータ処理装置１から強制的に抜かれてしまったりして、処理が中断されてしまった場合、再起動時に、第１のブートブロックをブートブロックとして使用することはできない。このときは、予備のブートブロックである第２のブートブロックに格納されているブートデータを読み出してメモリカード２を起動する。また、このときは、第２のブートブロックから読み出されたブートデータに基づいて、第１のブートブロックを再構築する。

【０１０９】そして、第１のブートブロックが更新されると、図８（ｃ）に示すように、第１のブートブロックの識別番号が１インクリメントされ、本例ではその値が「１１」とされる。この段階では、第１のブートブロックには新しいブートデータが格納され、第２のブートブロックには古いブートデータが格納された状態となる。

【０１１０】したがって、もしもこの段階で、メモリカード２がデータ処理装置１から強制的に抜かれてしまったりして、処理が中断されてしまった場合、再起動時には、第１のブートブロックに格納されている新しいブートデータ（すなわち、識別番号の値が大きいほうのブートブロックに格納されているブートデータ）を読み出してメモリカード２を起動する。また、このときは、識別番号の値が大きいほうのブートブロック（すなわち第１のブートブロック）に格納されているブートデータに基づいて、識別番号の値が小さいほうのブートブロック（すなわち第２のブートブロック）のブートデータを更新する。

【０１１１】そして、第１のブートブロックの更新が完了したら、次に、図８（ｄ）に示すように、物理アドレスの大きいほうのブートブロック、すなわち第２のブートブロックの更新を行う。このとき、第２のブートブロックは、ブートデータの書き換えの最中であるので、ブートブロックとしては使用できない「ＮＧ」状態となる。

【０１１２】もしもこの段階で、メモリカード２がデータ処理装置１から強制的に抜かれてしまったりして、処理が中断されてしまった場合、再起動時には、第１のブ

ートブロックのブートデータを使用してメモリカード2は起動されるが、予備のブートブロックが存在しない状態となる。したがって、このときは、第1のブートブロックから読み出されたブートデータに基づいて、予備のブートブロックである第2のブートブロックを再構築する。

【0113】そして、第2のブートブロックが更新されると、図8（e）に示すように、第2のブートブロックの識別番号が1インクリメントされ、本例ではその値が「11」とされる。これにより、第1のブートブロックの識別番号と、第2のブートブロックの識別番号とが等しい値となる。以上の処理により、第1のブートブロックにも第2のブートブロックにも新しいブートデータが格納された状態となる。

【0114】（3）メモリカード起動時のブートブロック読み込みの手順

ブートブロックに格納されたブートデータは、メモリカード2がデータ処理装置1に接続され起動されたときに、データ処理装置1に最初に読み込まれる。そこで、つぎに、メモリカード起動時における、データ処理装置1によるブートブロック読み込みの手順について詳細に説明する。

【0115】正常な状態では、同一識別番号を持ち且つ同一内容を持つブートブロックが2つ存在する。そこで、このことを確認する処理を、メモリカード起動時に必ず行う。具体的には、先頭ブロックから順番に、以下に示すような流れで処理を行い、ブートブロックを調べていく。

【0116】・先頭ページの分散管理情報が正常に読めるかを確認する。

【0117】・可／不可フラグが「使用可」になっているかを確認する。

【0118】・ブロックフラグが「先頭使用」になっているかを確認する。

【0119】・ブート識別子を検出して、格納されているデータがブートデータであるかを確認する。

【0120】・最終ページに格納されている追加管理情報が正常に読めるかを確認する。

【0121】・識別番号を読み出す。このとき、1個目のブートブロックの場合には、その値を保存する。2個目のブートブロックの場合には、1個目のブートブロックの識別番号と値が一致するかを確認する。

【0122】・ブートブロック内のデータを読み出す。1個目のブートブロックの場合、読み出したデータは保存しておく。2個目のブートブロックの場合、読み出したデータと保存しておいた1個目のブートブロックのデータとが一致するかを確認する。

【0123】以上のような処理を先頭ブロックから順次行い、同一識別番号を持ち且つ同一内容を持つブートブロックが2つ確認された時点で、ブートブロックの読み込みを終了する。このように、メモリカード2の起動時に2つのブートブロックの確認を行うことにより、メモリカード2の信頼性を非常に高めることができる。

【0124】なお、物理アドレスが上述した所定の値Mになるまで調べた時点で、ブートブロックが1つしか存在しない場合には、当該ブートブロックに格納されているブートデータを使用してメモリカード2を起動する。このとき、先頭からM番目までのブロックに使用可能なブロックがある場合には、当該ブロックにブートデータを書き込んで、改めて予備のブートブロックを作成する。先頭からM番目までのブロックに使用可能なブロックがない場合には、ブートブロックが1つの状態のままでメモリカード2を動作させる。また、M番目のブロックまで調べた時点で、ブートブロックが存在しなかった場合には、データ処理装置1はメディア異常と判断し、そのメモリカード2を受け付けないようにする。

【0125】なお、2つのブロックにそれぞれブートデータが正常に書かれていても識別番号が異なっている場合には、識別番号の大きいほうのブロックを正当なブートブロックとして選択し、メモリカード2の起動には、識別番号の大きいほうのブロックに格納されているブートデータを使用する。ただし、一方の識別番号が「255」で他方の識別番号が「0」の場合には、識別番号が「0」のほうのブロックを正当なブートブロックとして選択し、当該ブロックのブートデータを使用する。また、1つのブロックにだけブートデータが正常に書かれているときには、そのブロックがブートブロックとして使用される。

【0126】つぎに、以上のようなメモリカード起動時のブートブロック読み込みの手順について、図9乃至図12に示すフローチャートを参照して、更に詳細に説明する。なお、ここでは、変数としてI，WB，IDA，IDBを使用する。変数I，WBは、物理アドレスが入力される変数であり、変数IDA，IDBは、識別番号の値が入力される変数である。

【0127】メモリカードの起動時には、図9に示すように、先ず、ステップS1において、変数Iに「0」を代入する。また、変数IDA，IDBに「0」を代入する。次に、ステップS2へ進む。

【0128】ステップS2では、処理の対象となるブロックを、変数Iが示す物理アドレスのブロックとする。次に、ステップS3へ進む。

【0129】ステップS3では、現在処理の対象となっているブロックから、分散管理情報を読み出せるかを判別する。分散管理情報を読み出せれば、ステップS4へ進み、分散管理情報を読み出せなければ、ステップS25へ進む。

【0130】ステップS4では、現在処理の対象となっているブロックの可／不可フラグが「使用可」になっているかを判別する。「使用可」になっていれば、ステッ

プＳ５へ進み、「使用可」になっていなければ、ステップＳ２２へ進む。

【０１３１】ステップＳ５では、現在処理の対象となっているブロックのブロックフラグが「先頭使用」になっているかを判別する。「先頭使用」になっていれば、ステップＳ６へ進み、「先頭使用」になっていなければ、ステップＳ２４へ進む。

【０１３２】ステップＳ６では、現在処理の対象となっているブロックに格納されているデータに、ブート識別子が設定されているかを判別する。すなわち、当該データがブートデータであるかを判別する。ブートデータであれば、ステップＳ７へ進み、ブートデータでなければ、ステップＳ２８へ進む。

【０１３３】ステップＳ７では、現在処理の対象となっているブロックから、追加管理情報を読み出せるかを判別する。追加管理情報を読み出せれば、ステップＳ８へ進み、追加管理情報を読み出せなければ、ステップＳ２５へ進む。

【０１３４】ステップＳ８では、変数ＩＤＡに、現在処理の対象となっているブロックの識別番号の値を代入する。次に、ステップＳ９へ進む。

【０１３５】ステップＳ９では、現在処理の対象となっているブロックから、ブートデータを読み出せるかを判別する。ブートデータを読み出せれば、ステップＳ１０へ進み、ブートデータを読み出せなければ、ステップＳ２５へ進む。

【０１３６】ステップＳ１０では、現在処理の対象となっているブロックからブートデータを読み出し、当該ブートデータを保存する。次に、ステップＳ１１へ進む。

【０１３７】ステップＳ１１では、変数ＷＢに変数Ｉの値を代入するとともに、変数Ｉの値を１インクリメントする。次に、図１０のステップＳ１２へ進む。

【０１３８】ステップＳ１２では、処理の対象となるブロックを、変数Ｉが示す物理アドレスのブロックとする。次に、ステップＳ１３へ進む。

【０１３９】ステップＳ１３では、現在処理の対象となっているブロックから、分散管理情報を読み出せるかを判別する。分散管理情報を読み出せれば、ステップＳ１４へ進み、分散管理情報を読み出せなければ、ステップＳ３２へ進む。

【０１４０】ステップＳ１４では、現在処理の対象となっているブロックの可／不可フラグが「使用可」になっているかを判別する。「使用可」になっていれば、ステップＳ１５へ進み、「使用可」になっていなければ、ステップＳ２９へ進む。

【０１４１】ステップＳ１５では、現在処理の対象となっているブロックのブロックフラグが「先頭使用」になっているかを判別する。「先頭使用」になっていれば、ステップＳ１６へ進み、「先頭使用」になっていなければ、ステップＳ３１へ進む。

【０１４２】ステップＳ１６では、現在処理の対象となっているブロックに格納されているデータに、ブート識別子が設定されているかを判別する。すなわち、当該データがブートデータであるかを判別する。ブートデータであれば、ステップＳ１７へ進み、ブートデータでなければ、ステップＳ３５へ進む。

【０１４３】ステップＳ１７では、現在処理の対象となっているブロックから、追加管理情報を読み出せるかを判別する。追加管理情報を読み出せれば、ステップＳ１８へ進み、追加管理情報を読み出せなければ、ステップＳ３２へ進む。

【０１４４】ステップＳ１８では、変数ＩＤＢに、現在処理の対象となっているブロックの識別番号の値を代入する。次に、ステップＳ１９へ進む。

【０１４５】ステップＳ１９では、変数ＩＤＡの値と変数ＩＤＢの値とを比較する。変数ＩＤＡの値と変数ＩＤＢの値とが等しければ、ステップＳ２０へ進み、変数ＩＤＡの値と変数ＩＤＢの値とが等しくなければ、ステップＳ３２へ進む。

【０１４６】ステップＳ２０では、現在処理の対象となっているブロックから、ブートデータを読み出せるかを判別する。ブートデータを読み出せれば、ステップＳ２１へ進み、ブートデータを読み出せなければ、ステップＳ３２へ進む。

【０１４７】ステップＳ２１では、現在処理の対象となっているブロックからブートデータを読み出し、当該ブートデータと、先に読み出して保存しておいたブートデータとが一致しているかを判別する。それらのブートデータが一致しているのは、２つのブートブロックからのブートデータの読み出しが正常に行われたときであるので、これで処理を終了する。一方、それらのブートデータが一致していなければ、ステップＳ３２へ進む。

【０１４８】一方、図９に示すように、ステップＳ４で可／不可フラグが「使用可」になっていなかった場合には、上述したようにステップＳ２２へ進む。

【０１４９】ステップＳ２２では、変数Ｉの値と、上述した所定の値Ｍとを比較する。そして、変数Ｉの値が所定の値Ｍよりも小さければ、ステップＳ２３へ進む。ステップＳ２２において、変数Ｉの値が所定の値Ｍ以上となるのは、Ｍ番目のブロックまで調べてもブートデータが得られなかった場合であり、このときは、エラーとして処理を終了する。

【０１５０】ステップＳ２３では、変数Ｉの値を１インクリメントする。そして、ステップＳ２へ戻って処理を繰り返す。

【０１５１】また、ステップＳ５でブロックフラグが「先頭使用」になっていなかった場合は、上述したようにステップ２４へ進む。このステップＳ２４では、当該ブロックフラグが「未使用」になっているかを判別する。「未使用」になっていれば、図１１のステップＳ３

６へ進み、「未使用」になっていなければ、ステップ２５へ進む。

【０１５２】ステップＳ２５では、現在処理の対象となっているブロックに対して消去処理を施す。次に、ステップＳ２６へ進む。

【０１５３】ステップＳ２６では、ステップＳ２５での消去処理が正常に完了したかを判別する。消去処理が正常に完了していれば、図１１のステップＳ３６へ進み、消去処理が正常に完了していなければ、ステップ２７へ進む。

【０１５４】ステップＳ２７では、現在処理の対象となっているブロックの可／不可フラグを「使用不可」に設定し、その後、ステップＳ２２へ進んで上述した処理を行う。

【０１５５】また、ステップＳ６でデータがブートデータでなかった場合には、上述したようにステップＳ２８へ進む。

【０１５６】このステップＳ２８では、処理の対象となるブロックを他のブロックに移動する。その後、ステップＳ２５へ進んで上述した処理を行う。

【０１５７】また、ステップＳ３で分散管理情報が読み出せなかった場合、ステップＳ７で追加管理情報を読み出せなかった場合、及びステップＳ９でブートデータを読み出せなかった場合にも、上述したようにステップＳ２５へ進んで、上述した処理を行う。

【０１５８】また、図１０に示すように、ステップＳ１４で可／不可フラグが「使用可」になっていなかった場合には、上述したようにステップＳ２９へ進む。このステップＳ２９では、変数Ｉの値と、上述した所定の値Ｍとを比較する。そして、変数Ｉの値が所定の値Ｍよりも小さければ、ステップＳ３０へ進む。一方、ステップＳ２９において変数Ｉの値が所定の値Ｍ以上となるのは、Ｍ番目のブロックまで調べても２つめのブートデータが得られなかった場合であり、このときは、ここでブートデータ読み出しの処理を終了し、先に読み出して保存しておいたブートデータを使ってメモリカード２を起動する。

【０１５９】ステップＳ３０では、変数Ｉの値を１インクリメントする。そして、ステップＳ１２へ戻って処理を繰り返す。

【０１６０】また、ステップＳ１５でブロックフラグが「先頭使用」になっていなかった場合は、上述したようにステップ３１へ進む。このステップＳ３１では、当該ブロックフラグが「未使用」になっているかを判別する。「未使用」になっていれば、図１２のステップＳ４７へ進み、「未使用」になっていなければ、ステップ３２へ進む。

【０１６１】ステップＳ３２では、現在処理の対象となっているブロックに対して消去処理を施す。次に、ステップＳ３３へ進む。

【０１６２】ステップＳ３３では、ステップＳ３２での消去処理が正常に完了したかを判別する。消去処理が正常に完了していれば、図１２のステップＳ４７へ進み、消去処理が正常に完了していなければ、ステップ３４へ進む。

【０１６３】ステップＳ３４では、現在処理の対象となっているブロックの可／不可フラグを「使用不可」に設定し、その後、ステップＳ２９へ進んで上述した処理を行う。

【０１６４】また、ステップＳ１６でデータがブートデータでなかった場合には、上述したようにステップＳ３５へ進む。このステップＳ３５では、処理の対象となるブロックを他のブロックに移動する。その後、ステップＳ３２へ進んで上述した処理を行う。

【０１６５】また、ステップＳ１３で分散管理情報が読み出せなかった場合、ステップＳ１７で追加管理情報を読み出せなかった場合、ステップＳ１９で変数ＩＤＡの値と変数ＩＤＢの値とが一致しなかった場合、ステップＳ２０でブートデータを読み出せなかった場合、及びステップＳ２１でブートデータが一致しなかった場合にも、上述したようにステップＳ３２へ進んで、上述した処理を行う。

【０１６６】また、図１１に示すように、ステップＳ３６では、変数ＷＢに変数Ｉの値を代入するとともに、変数Ｉの値を１インクリメントする。次に、ステップＳ３７へ進む。

【０１６７】ステップＳ３７では、処理の対象となるブロックを、変数Ｉが示す物理アドレスのブロックとする。次に、ステップＳ３８へ進む。

【０１６８】ステップＳ３８では、現在処理の対象となっているブロックから、分散管理情報を読み出せるかを判別する。分散管理情報を読み出せれば、ステップＳ３９へ進み、分散管理情報を読み出せなければ、ステップＳ４５へ進む。

【０１６９】ステップＳ３９では、現在処理の対象となっているブロックの可／不可フラグが「使用可」になっているかを判別する。「使用可」になっていれば、ステップＳ４０へ進み、「使用可」になっていなければ、ステップＳ４５へ進む。

【０１７０】ステップＳ４０では、現在処理の対象となっているブロックに格納されているデータに、ブート識別子が設定されているかを判別する。すなわち、当該データがブートデータであるかを判別する。ブートデータであれば、ステップＳ４１へ進む。

【０１７１】ステップＳ４１では、現在処理の対象となっているブロックから、追加管理情報を読み出せるかを判別する。追加管理情報を読み出せれば、ステップＳ４２へ進む。

【０１７２】ステップＳ４２では、現在処理の対象となっているブロックから、ブートデータを読み出せるかを

判別する。ブートデータを読み出せれば、ステップＳ４３へ進む。

【０１７３】ステップＳ４３では、現在処理の対象となっているブロックからブートデータを読み出し、当該ブートデータを保存する。次に、ステップＳ４４へ進む。

【０１７４】ステップＳ４４では、変数ＷＢが示す物理アドレスのブロックに、ステップＳ４３で読み出して保存しておいたブートデータを書き込む。このときは、ここでブートデータ読み出しの処理を終了し、ステップＳ４３で読み出して保存しておいたブートデータを使ってメモリカード２を起動する。

【０１７５】また、ステップＳ３８で分散管理情報が読み出せなかった場合、及びステップＳ３９で可／不可フラグが「使用可」になっていなかった場合には、上述したようにステップＳ４５へ進む。このステップＳ４５では、変数Ｉの値と、上述した所定の値Ｍとを比較する。そして、変数Ｉの値が所定の値Ｍよりも小さければ、ステップＳ４６へ進む。ステップＳ４５において、変数Ｉの値が所定の値Ｍ以上となるのは、Ｍ番目のブロックまで調べてもブートデータが得られなかった場合であり、このときは、エラーとして処理を終了する。

【０１７６】ステップＳ４６では、変数Ｉの値を１インクリメントする。そして、ステップＳ３７へ戻って処理を繰り返す。

【０１７７】また、ステップＳ４０でデータがブートデータではなかった場合、ステップＳ４１で追加管理情報を読み出せなかった場合、及びステップＳ４２でブートデータを読み出せなかった場合には、エラーとして処理を終了する。これは、２つのブートブロックのいずれからもブートデータが読み出せなかった場合である。

【０１７８】また、図１２に示すように、ステップＳ４７では、変数ＩＤＢの値が「0xffff」であるかを判別する。変数ＩＤＢの値が「0xffff」でなければステップＳ４８へ進み、変数ＩＤＢの値が「0xffff」であればステップＳ４９へ進む。

【０１７９】ステップＳ４８では、変数ＩＤＡの値と、変数ＩＤＢの値に「１」を加えた値とを比較する。これらの値が等しければ、ステップＳ４９へ進み、等しくなければ、ステップＳ５１へ進む。

【０１８０】ステップＳ４９では、変数ＷＢに変数Ｉの値を代入する。次に、ステップＳ５０へ進む。

【０１８１】ステップＳ５０では、変数ＷＢが示す物理アドレスのブロックに、先に読み出して保存しておいたブートデータを書き込む。このときは、ここでブートデータ読み出しの処理を終了し、先に読み出して保存しておいたブートデータを使ってメモリカード２を起動する。

【０１８２】ステップＳ５１では、変数ＩＤＢの値と、変数ＩＤＡの値に「１」を加えた値とを比較する。これらの値が等しければ、ステップＳ５０へ進み上述した処理を行う。

【０１８３】一方、ステップＳ５１において、変数ＩＤＢの値と、変数ＩＤＡの値に「１」を加えた値とが等しくないと判別されるのは、２つのブートブロックの識別番号が、等しくなく且つ連続番号でもない場合である。このような状態となるのは、メモリカード２に何らかの異常があったときである。しかしながら、ブートデータが読み出せなくなっているわけではないので、このときは、例えば手動復旧モードとして、データ処理装置１の側で適切な処理を行うようにする。

【０１８４】以上がメモリカード起動時のブートブロック読み込みの手順である。このように手順により、メモリカード２の起動時に２つのブートブロックの確認を行うことで、メモリカード２の信頼性を非常に高めることができる。

【０１８５】

【発明の効果】以上詳細に説明したように、本発明によれば、複数の異なるブロックにブートデータをそれぞれ格納するようにしているので、たとえブートデータが格納されたブロックのうちのいずれかが使用不可能になったとしても、他のブロックのブートデータを用いて、外部記憶装置を起動することができる。

【０１８６】しかも、それらのブロックにはブートデータの新旧を示す識別番号を格納するようにしているので、常に最新のブートデータを使用するようにすることができる。すなわち、本発明によれば、たとえ新しいブートデータと古いブートデータとが混在するような状態になったとしても、常に最新のブートデータを使用するようにすることができ、データの一貫性を保証できる。

【０１８７】したがって、本発明によれば、ブートデータ読み出し時にエラーが発生し難く、信頼性の高い外部記憶装置を提供することができる。

【図面の簡単な説明】

【図１】本発明が適用されるシステムの全体構成を示す図である。

【図２】本発明を適用したメモリカードの構成を示すブロック図である。

【図３】本発明を適用したメモリカードの外観を示す斜視図である。

【図４】本発明を適用したメモリカードの記憶領域の構造を示す図である。

【図５】分散管理情報の構成を示す図である。

【図６】追加管理情報の構成を示す図である。

【図７】各ブロックの分散管理情報から集合管理情報を構築する様子を示す図である。

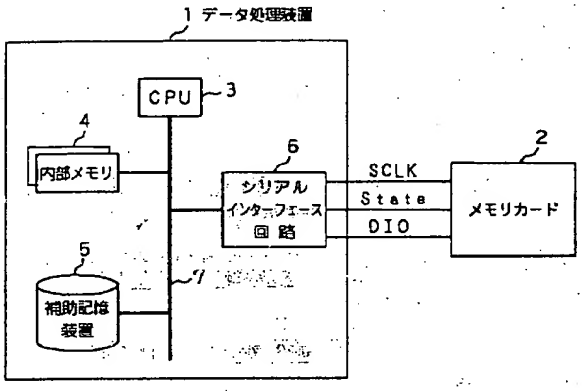
【図８】ブートブロックを更新する際の手順を示す図である。

【図９】メモリカード起動時のブートブロック読み込みの手順を示すフローチャートである。

【図１０】メモリカード起動時のブートブロック読み込

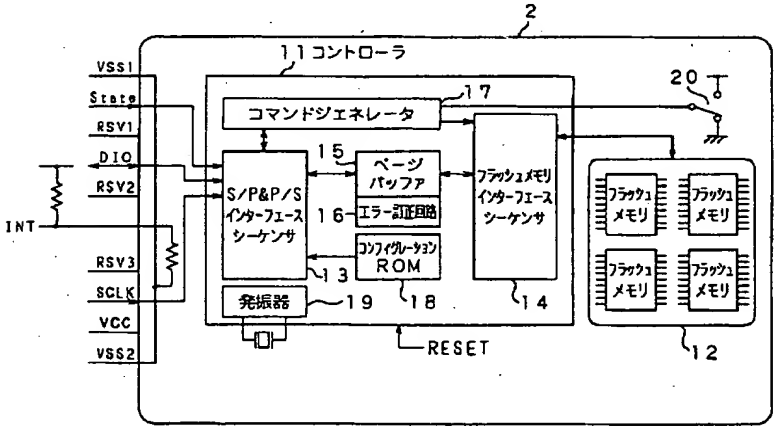
みの手順を示すフローチャートである。
【図11】メモリカード起動時のブートブロック読み込みの手順を示すフローチャートである。
【図12】メモリカード起動時のブートブロック読み込みの手順を示すフローチャートである。
【符号の説明】
1　データ処理装置、　2　メモリカード、　3　演算処理装置、　4　内部メモリ、　5　補助記憶装置、　6　シリアルインターフェース回路、　7　バス、　11　コントローラ、　12　フラッシュメモリ、　13　シリアル／パラレル・パラレル／シリアル・インターフェース・シーケンサ、　14　フラッシュメモリ・インターフェース・シーケンサ、　15　ページバッファ、　16エラー訂正回路、　17　コマンドジェネレータ、　18　コンフィグレーションROM、　19　発振器、　20　誤消去防止スイッチ

【図1】



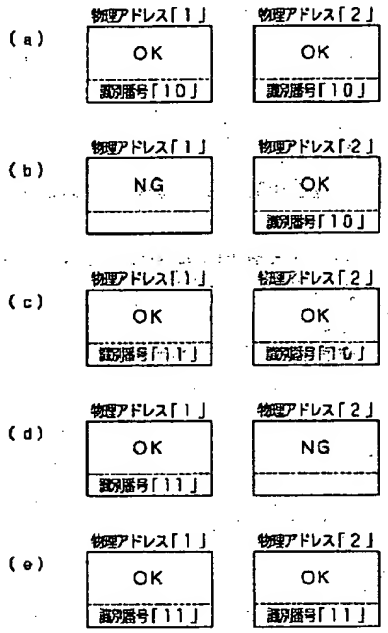
システムの全体構成

【図2】



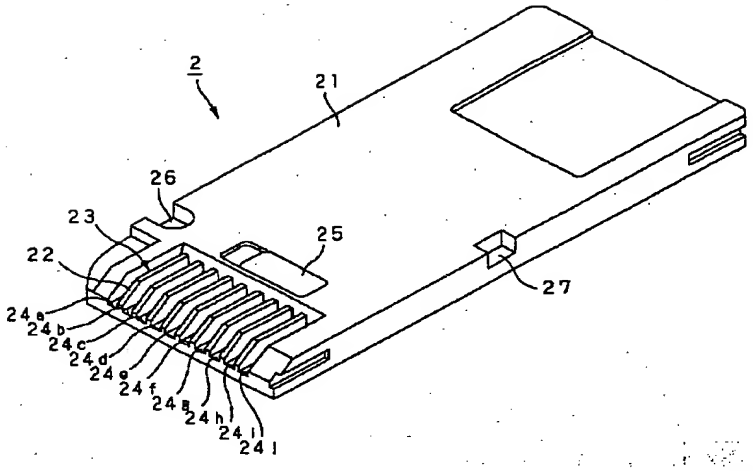
メモリカードの構成

【図8】

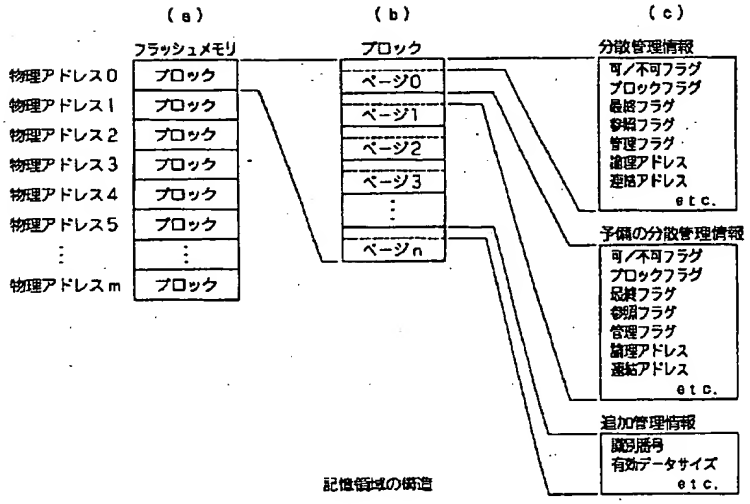


ブートブロックの更新時の手順

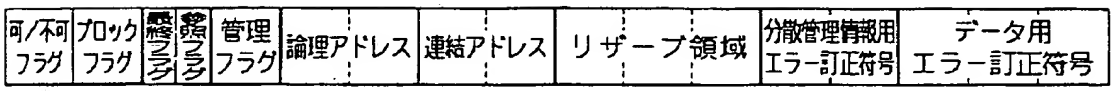
【図3】


2
21
26
23
22
25
27
24a
24b
24c
24d
24e
24f
24g
24h
24i
24j

【図4】

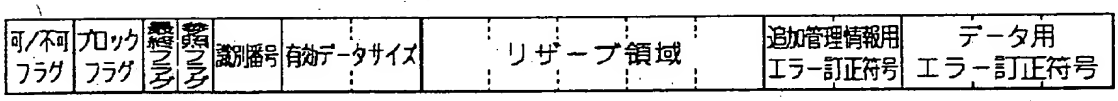

(a)
(b)
(c)
フラッシュメモリ
ブロック
分散管理情報
物理アドレス0
物理アドレス1
物理アドレス2
物理アドレス3
物理アドレス4
物理アドレス5
物理アドレスm
ブロック
ページ0
ページ1
ページ2
ページ3
ページn
可/不可フラグ
ブロックフラグ
最終フラグ
参照フラグ
管理フラグ
論理アドレス
連結アドレス
etc.
予備の分散管理情報
追加管理情報
識別番号
有効データサイズ
記憶領域の構造

【図5】



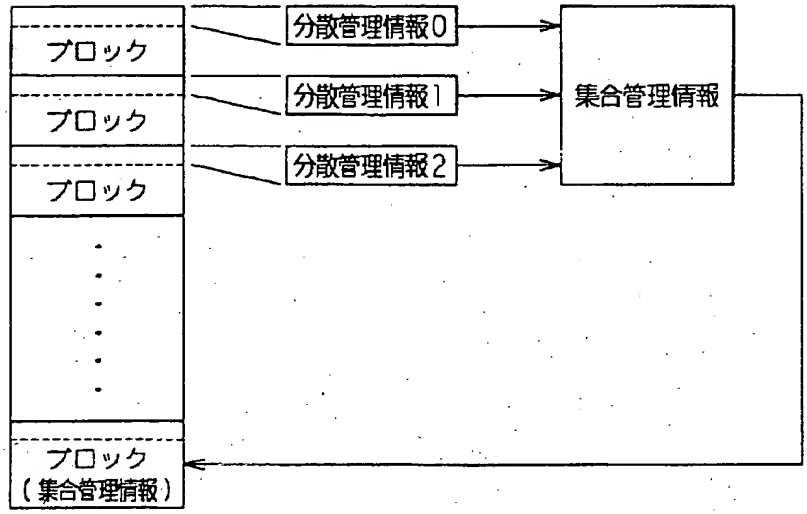
分散管理情報

【図6】



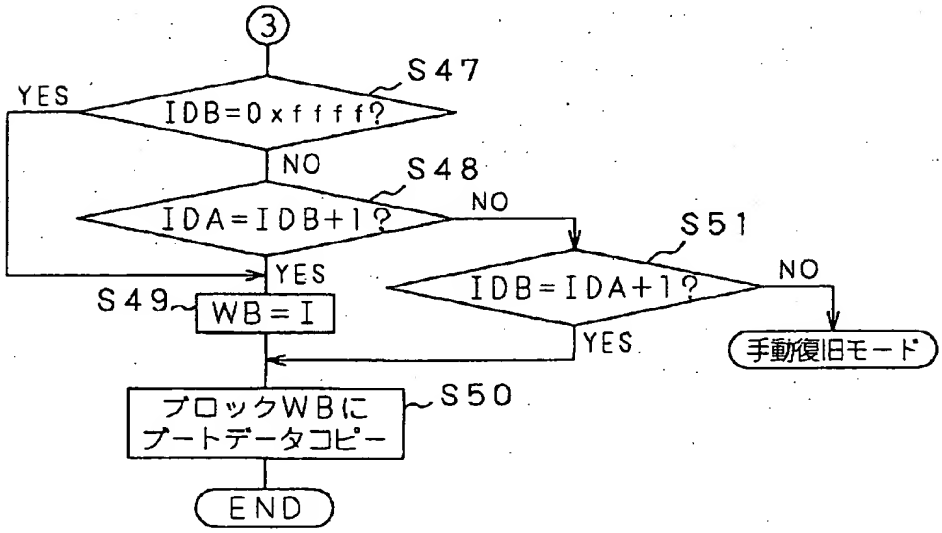
追加管理情報

【図7】



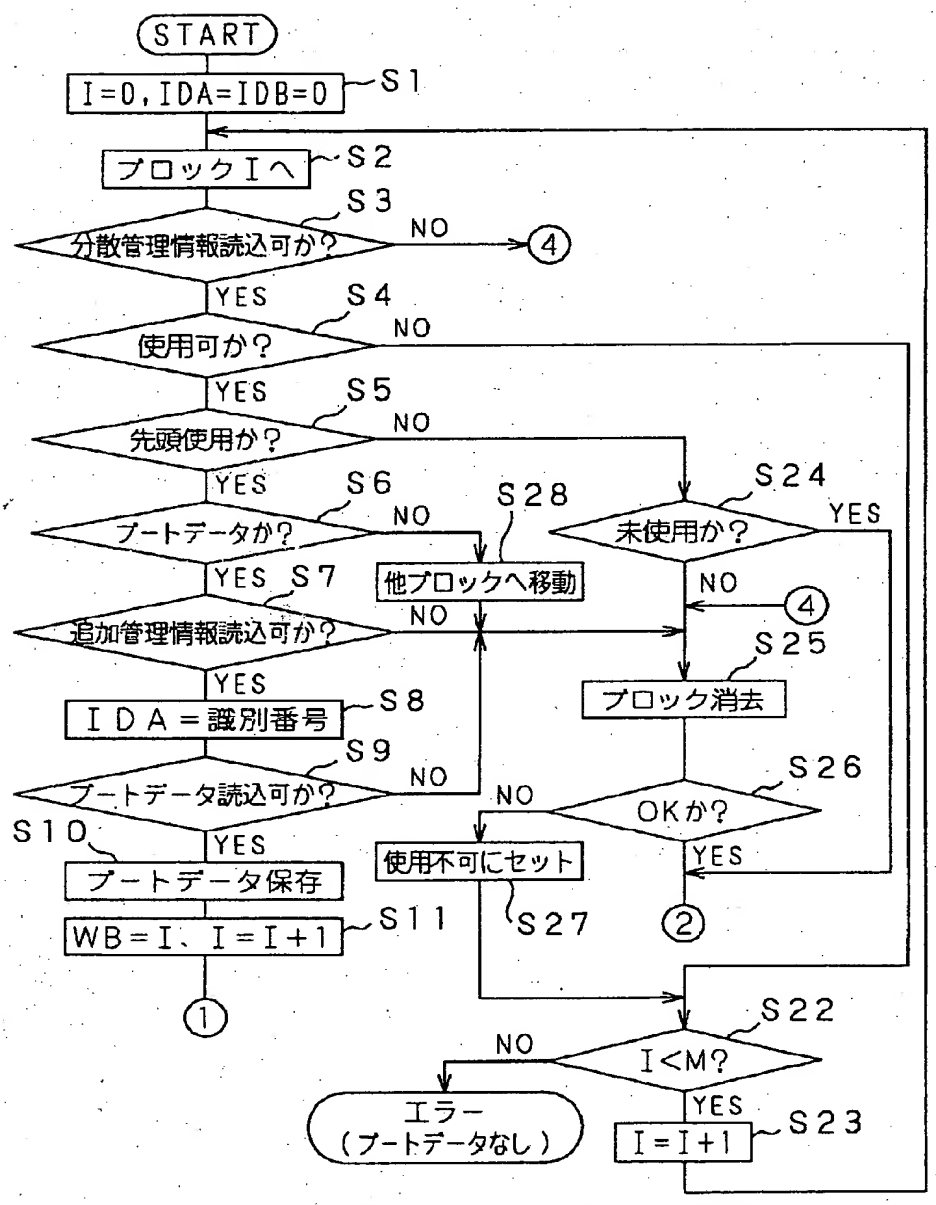
分散管理情報から集合管理情報の構築

【図12】



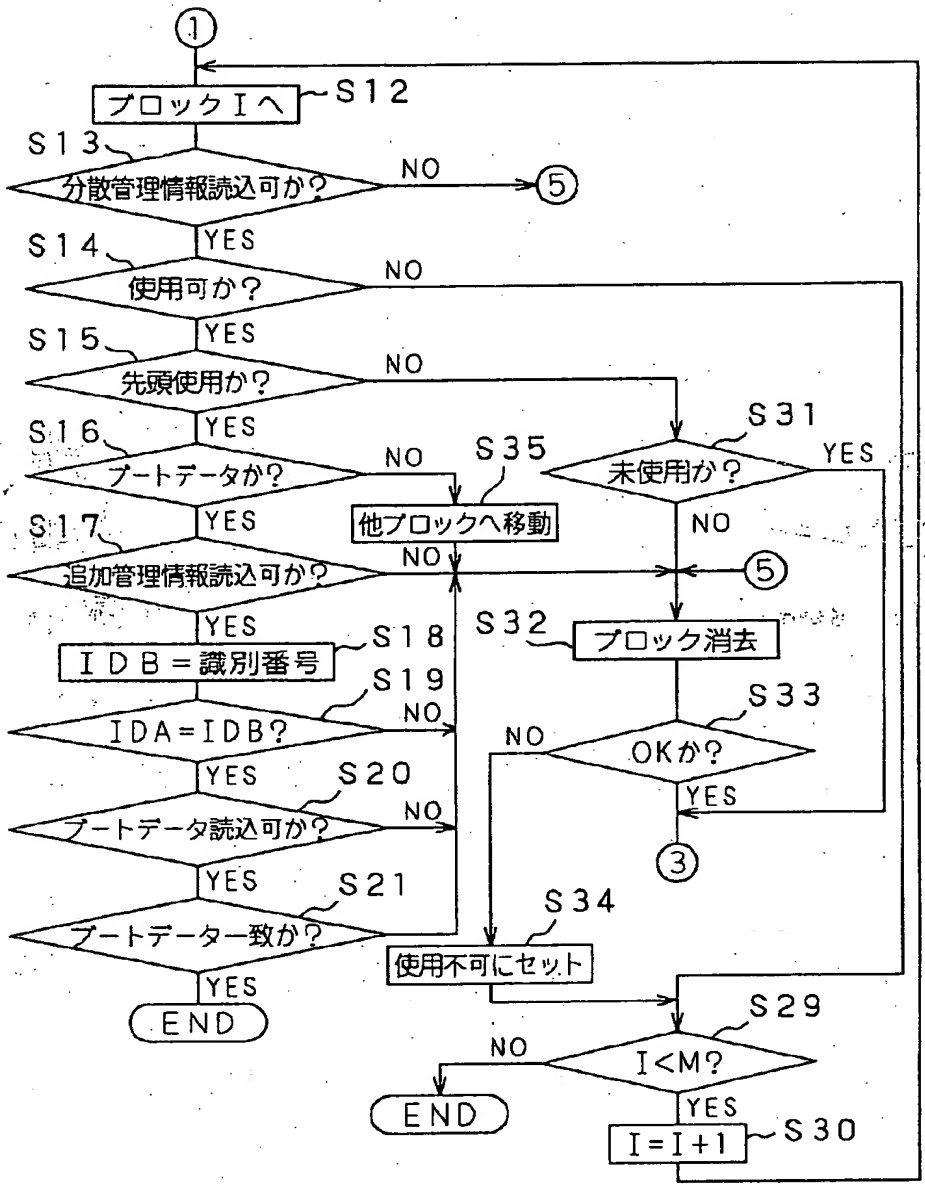
メモリカード起動時のブートブロック読み込みの手順

【図９】



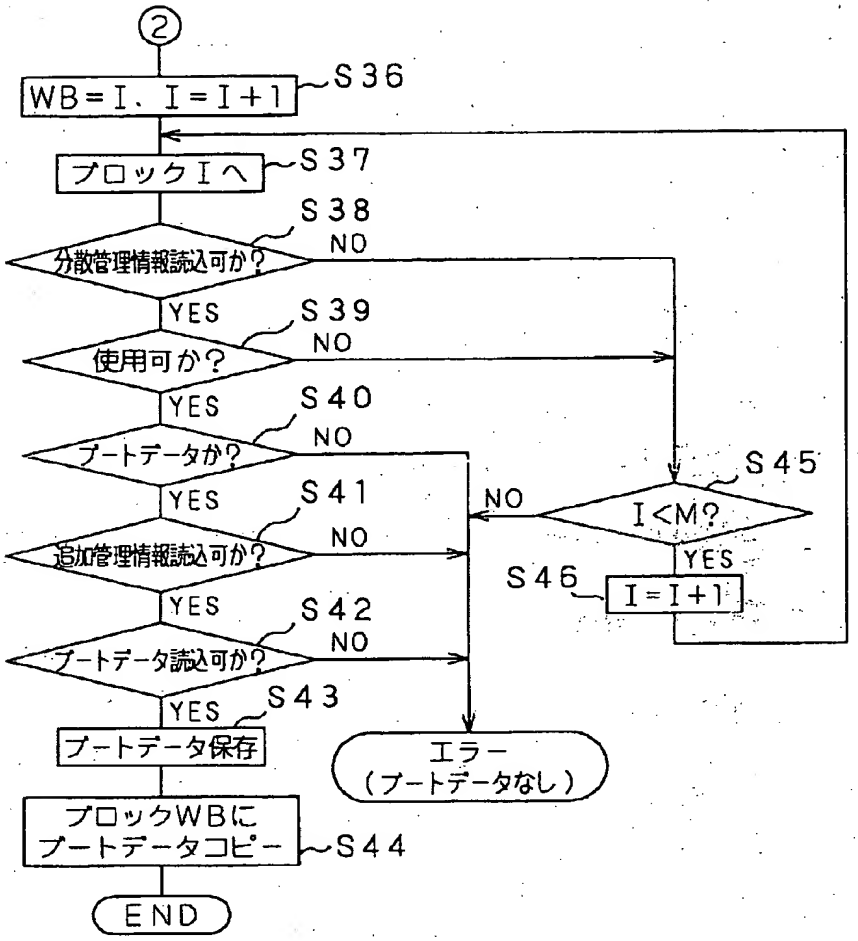
メモリカード起動時のブートブロック読み込みの手順

【図10】



メモリカード起動時のブートブロック読み込みの手順

【図11】



メモリカード起動時のブートブロック読み込みの手順

【手続補正書】
【提出日】平成１１年１月２６日
【手続補正１】
【補正対象書類名】明細書
【補正対象項目名】特許請求の範囲
【補正方法】変更
【補正内容】
【特許請求の範囲】
【請求項１】　データの消去を所定のブロック単位で行うとともに、起動時に最初に読み出されるブートデータを格納するブロックを有する外部記憶装置であって、
ブートデータがそれぞれ格納された複数の異なるブロックを備えるとともに、
ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号が格納される識別番号格納部を備えていることを特徴とする外部記憶装置。
【請求項２】　データの消去を所定のブロック単位で行う外部記憶装置に、当該外部記憶装置の起動時に最初に読み出されるブートデータを格納する際に、
複数の異なるブロックにブートデータをそれぞれ格納するとともに、
ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納することを特徴とするデータ処理方法。
【請求項３】　上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納

された各ブートデータのうち、最新のブートデータを読み出して、当該ブートデータを用いて外部記憶装置を起動することを特徴とする請求項２記載のデータ処理方法。

【請求項４】　上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータの新旧を判別し、古いブートデータがある場合には当該ブートデータを最新のブートデータに書き換えることを特徴とする請求項２記載のデータ処理方法。

【手続補正２】

【補正対象書類名】明細書

【補正対象項目名】００１０

【補正方法】変更

【補正内容】

【００１０】

【課題を解決するための手段】本発明に係る外部記憶装置は、データの消去を所定のブロック単位で行うとともに、起動時に最初に読み出されるブートデータを格納するブロックを有する外部記憶装置である。そして、ブートデータがそれぞれ格納された複数の異なるブロックを備えるとともに、ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号が格納される識別番号格納部を備えていることを特徴としている。